Carbon-Aware Cloud Application

Nan Deng
The Ohio State University

Student Name Nan Deng

Email dengn@cse.ohio-state.edu

Degree of Study Ph.D

A growing cadre of green datacenters now use rooftop solar panels and on-site wind turbines [3, 2], to reduce their carbon footprint. Increasingly, individual applications within the datacenter are under pressure to reduce their own carbon footprints by using such renewable-powered servers. For example, over 500,000 Facebook members have signed a petition pressuring the site's managers to use only renewable energy [4].

This demo shows the implementation of carbon-aware Apache, a distributed cloud application that manages its carbon emissions via dynamic provisioning. Carbon-aware Apache keeps its carbon footprint below a preset value by monitoring how much renewable energy powers the servers it uses. Unused servers are turned off. We use the term *cloud instances* to refer to servers leased for a fixed period of time. In our demo, we control the emissions costs of these instances, by simulating the production of renewable energy. We expose emissions costs to carbon-aware Apache's provisioning mechanism, which decides which cloud instances to provision such that 1) the carbon limit (measured in terms of grid energy) is not exceeded and 2) performance goals are met. In this demo, we focus on throughput (i.e., requests per second or job processing rate) as a performance metric. We will show our infrastructure for distributing renewable energy and tracking it within a compute cluster.

We beleive that renewable-powered distributed systems are a new and emerging application that many NOMS attendees will want to see in action. Attendees will be able to interact with carbon-aware Apache, setting the request arrival rate, renewable-energy production, and carbon-aware policy. We will set up a video display that shows which servers are used and the total throughput.

Mathematical Model: Suppose there are n cloud instances available for provisioning. At time period t, a provisioning strategy for an application is denoted as a vector $\mathbf{x^{(t)}} = (x_1^{(t)}, x_2^{(t)} \cdots x_n^{(t)})^T$, in which $x_i^{(t)} = 1$ if the ith instance is provisioned, otherwise $x_i^{(t)} = 0$. Each cloud instance can provide up to its *maximum throughput* for a target application, represented by the vector $\mathbf{v} = (v_1, v_2 \cdots v_n)^T$. v_i is a real number that denotes the maximum throughput provided by the server which will run ith instance. The *maximum throughput* of a cloud application is the summation of the maximum throughput of its provisioned instances, i.e. $\mathbf{v}^T\mathbf{x^{(t)}}$. Here, the symbol T is the transpose function in vector multiplication. We also note that some applications, e.g., Web servers, may have fluctuating throughput needs over time. We use $V^{(t)}$ to represent the target throughput at time t.

The hard limit on carbon-heavy, grid energy is $D^{(t)}$ as set by the application manager. The total grid energy used by an instance at time t is represented as a vector $\mathbf{d^{(t)}} = (d_1^{(t)}, d_2^{(t)} \cdots d_n^{(t)})^T$. Note, $d_i^{(t)}$ is a real number between zero and max energy needs of the ith instance. An application's total grid energy consumption at time t is $\mathbf{d}^T\mathbf{x^{(t)}} < D^{(t)}$.

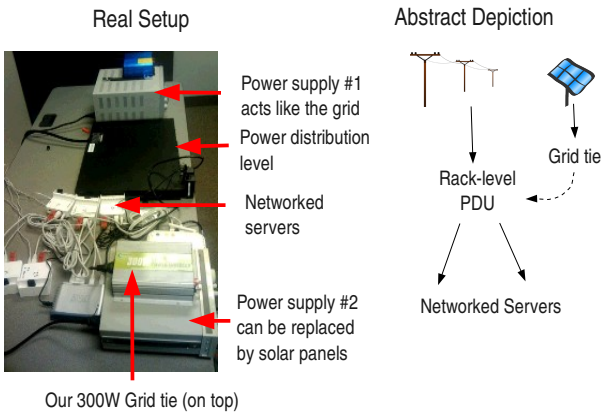

Figure 1: Our renewable-energy cluster; the real setup and abstract goal. Dotted lines reflect grid-tie placements.

Given $\mathbf{v}, \mathbf{d^{(t)}}, D^{(t)}, V^{(t)}$, our goal is to find $\mathbf{x^{(t)}}$. Specifically, we represent carbon-aware provisioning as an integer programming problem:

$$\text{Maximize } \mathbf{v}^T\mathbf{x^{(t)}} \tag{1}$$

$$\text{Subject to } \mathbf{d^{(t)}}^T\mathbf{x^{(t)}} \leq D^{(t)} \tag{2}$$

$$\text{and } \mathbf{v}^T\mathbf{x^{(t)}} \leq V^{(t)} \tag{3}$$

Implementation: Figure 1 shows an early (less portable) version of our demo system, along with its abstraction in a real datacenter. The demo consists of several ARM-based low-power GNU/Linux servers, each with Apache web server. During demo, we will use 5 servers making the whole system's maximum power consumption about 200 Watts. Importantly, this keeps the power consumption well within standard 120V, 15-AMP US regulations. The system's network is organized through a 1GbE router and a 1GbE switch. The power supply of the system consists of both carbon-heavy power and clean power. As shown in the figure, the top component is a power supply unit with capacity that easily exceeds the max power of our cluster—it simulates the electric grid. The bottom component is a programmable power supply that connects to a grid tie; it can be replaced by a real solar panel on a sunny day. All of these components operate around only 5% of their nameplate power capacity. We will also need power outlets to support the display system (not shown). During the demo, attendees will input parameters to the mathematical model above. The output will be forwarded to the power controls on the PDU. The PDU will turn off unused servers, while the others complete web requests. We we use the scaled WorldCup[1] traces.

References

[1] M. Arlitt and T. Jin. A workload characterization study of the 1998 world cup web site. *IEEE Network*, 14(3), 2000.

[2] Environmental Leader. Data centers power up savings with renewable energy. `http://www.environmentalleader.com/2009/07/29/data-centers-power-up-savings-with-renewable-energy/`.

[3] Green House Data: Greening the data center. `http://www.greenhousedata.com/`.

[4] G. USA. Facebook status update: Renewable energy now. `http://www.greenpeace.org/usa/news/facebook-update-renewable-ene`.